University of Southern California

2

A Distributed Architecture for Multimedia Conference Control

Eve M. Schooler

ISI/RR-91-289
November 1991

INFORMATION SCIENCES INSTITUTE ISI 213/822-1511
4676 Admiralty Way/Marina del Rey/California 90292-6695

A Distributed Architecture for Multimedia Conference Control

Eve M. Schooler

ISI/RR-91-289
November 1991

University of Southern California
Information Sciences Institute
4676 Admiralty Way, Marina del Rey, CA 90292-6695
310-822-1511

This research was sponsored by the Defense Advanced Research Projects Agency under contract number DABT63-91-C-0001. Views and conclusions contained in this report are the authors' and should not be interpreted as representing the official opinion or policy of DARPA, the U.S. Government, or any person or agency connected with them.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			
4. PERFORMING ORGANIZATION REPORT NUMBER(S) ISI/RR-91-289		5. MONITORING ORGANIZATION REPORT NUMBER(S)	
6a. NAME OF PERFORMING ORGANIZATION USC/Information Sciences Institute	6b. OFFICE SYMBOL	7a. NAME OF MONITORING ORGANIZATION Department of the Army	
6c. ADDRESS *(City, State, and ZIP Code)* 4676 Admiralty Way Marina del Rey, CA 90292-6695		7b. ADDRESS *(City, State, and ZIP Code)* Directorate of Contracting, ATTN: ATZS-DKM Post Office Box 748 Fort Huachuca, AZ 85613-0748	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION DARPA	8b. OFFICE SYMBOL	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER DARPA (Fort Huachuca) DABT63-91-C-0001	
8c. ADDRESS *(City, State, and ZIP Code)* 3701 N. Fairfax Drive Arlington, VA 22203-1714		10. SOURCE OF FUNDING NUMBERS	

10. PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
---	---	---	---

11. TITLE *(Include Security Classification)*
A Distributed Architecture for Multimedia Conference Control (Unclassified)

12. PERSONAL AUTHOR(S)
Schooler, E.M.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(y/m/d)*	15. PAGE COUNT
Research Report	FROM______ TO______	1991, November	22

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP
09	02	

18. SUBJECT TERMS *(Use reverse if needed; identify by block no.)*
Multimedia, teleconferencing, connection management, configuration management, collaborative work, groupware

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

MMCC, the multimedia conference control program, is a window-based tool for conference management. It serves as an application interface to the ISI/BBN teleconferencing system, where it is used not only to orchestrate multisite conferences, but also to provide local and remote audio and video control, and to interact with other conference-oriented tools that support shared workspaces. The motivation for this paper is to document the design, operation and continued evolution of MMCC. After presenting the context for this work, we provide a discussion of MMCC's peer-to-peer model of communication and an overview of its connection control protocol. Issues are also raised about heterogeneity, robust services, scalability and the impact of conferencing over the Internet. A description of the system's regular use offers insights into the feasibility of the architecture. Finally, future directions for research in multimedia conference control are presented.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT [X]Unclassified/Unlimited [ ]Same as Rpt [ ]DTIC Users	21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL(S) Kathleen McLaughlin/Celeste Anderson	22b. TELEPHONE (310) 822-1511	22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Unclassified

A Distributed Architecture for Multimedia Conference Control

Eve M. Schooler
USC/Information Sciences Institute
schooler@isi.edu
November 1991

Abstract

MMCC, the multimedia conference control program, is a window-based tool for conference management. It serves as an application interface to the ISI/BBN teleconferencing system, where it is used not only to orchestrate multisite conferences, but also to provide local and remote audio and video control, and to interact with other conference-oriented tools that support shared workspaces. The motivation for this paper is to document the design, operation and continued evolution of MMCC. After presenting the context for this work, we provide a discussion of MMCC's peer-to-peer model of communication and an overview of its connection control protocol. Issues are also raised about heterogeneity, robust services, scalability and the impact of conferencing over the Internet. A description of the system's regular use offers insights into the feasibility of the architecture. Finally, future directions for research in multimedia conference control are presented.

1. Introduction

The integration of real-time, multimedia data into traditional computer workstation architectures is underway. The beginnings of this transition are already taking place with the availability of integrated audio in several scientific workstations and the recent announcement of built-in video capabilities such as JPEG chips, for still video images, and plans for MPEG chips, for motion video, to follow [11].

Making voice and video standard data types promises to challenge the way we design applications and our expectations from them. Current support for real-time data at all levels in the system is embryonic at best. There are neither standards to convey real-time requirements to various system interfaces, nor standards for collaborative use of the media. In addition, there is much work to be done to accommodate heterogeneity, resource sharing, and scalability.

While other researchers have concentrated on supporting real-time media from the user interface and operating system perspectives, the Multimedia Conferencing project (MMC), a joint effort between ISI and BBN, has focused primarily on the network and communications aspects. Toward this end, we have developed both a suite of experimental packet protocols [28, 6, 7] to provide performance guarantees for real-time data, and an underlying network, the Terrestrial Wideband Network (TWBnet) as an experimental testbed [3]. More recently, we have also begun using the DARPA Research Testbed (DARTnet) as a means for testing these protocols. As proof of concept, a real-time, multimedia conferencing system has been implemented that couples real-time packet-switched voice and video with a shared workspace, allowing geographically separated individuals to collaborate [23]. Teleconferencing sites are currently located near Los Angeles, San Francisco, Boston, Washington, D.C., Los Alamos, New Mexico, and Rome, New York, as well as in London, England.

MMCC, the multimedia conference control program, serves as the application interface to the teleconferencing system. Original design goals were to allow novice users to establish multimedia conferences, to allow participants to come and go from conferences as desired, and to make local and remote resources readily accessible and configurable. MMCC is a window-based application that automates multimedia conference establishment by hiding many of the underlying system details. Although it is used primarily for connection management, it also provides both local and remote control over real-time audio and video resources, and interacts with other conference-related tools (otherwise known as groupware) that support shared workspaces.

This paper focuses on our experiences with the design and operation of MMCC, first by discussing the conferencing system in general and then MMCC in particular. It highlights MMCC's peer-to-peer model of communication and gives an overview of the connection control protocol. The relationships between conference management and various system components, such as those that handle real-time multimedia processing and the RPC-based servers that mask the specifics of various multimedia devices, are presented in detail. Issues are also raised about the impact of conferencing over the Internet: heterogeneity, robust service, and scalability. A description of the system's regular use and continued evolution offer insights into the feasibility of the architecture. Finally, future directions for research in multimedia conference control are proposed.

2. System Overview

At present, the conferencing system might be described as providing coordinated management of separate services (audio, video, and groupware). Its general organization is depicted in Figure 1:

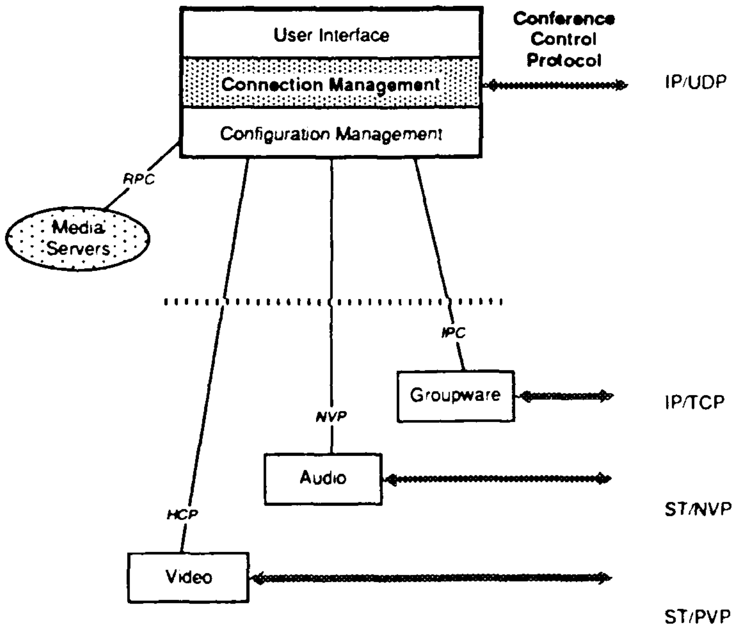


Figure 1. Coordinated Management of Separate Services

Conferences are orchestrated by MMCC, which plays essentially three roles. First and foremost, MMCC supplies connection management in the form of session establishment, maintenance and disconnection; it performs these tasks by communicating remotely with peer MMCCs using a connection control protocol and then by communicating locally with separate programs that actually handle the voice, video, and groupware connections. Second, MMCC presents a user interface to the underlying system components. Third, MMCC provides a configuration management interface to the various media. It is at this interface that we ask questions such as, "Do certain devices or individual media exist at a site? Are they available? Can they be configured as requested?" To assist with configuration management, MMCC interacts with several local RPC-based media servers, two of which are video-specific and one of which impacts both audio and video.

In the current implementation of the system, the video, audio and groupware components (called agents) are distinct from each other. They are also distinct from MMCC. Protocols have been devised to pass control information from MMCC to these subsystems, but MMCC leaves actual data flow to the underlying media-specific components themselves. This allows each agent to select an appropriate transport and internet level communication protocol when communicating with its counterpart at a remote participant's site, a protocol that best meets the needs of the data, be it real-time or non-real-time, control or non-control data. Details about the various agents and their protocols and interfaces are described in the upcoming sections.

3. Connection Control Protocol

Conference control is achieved through use of a peer-to-peer, distributed model. The assumption is that peer MMCCs reside on machines scattered throughout the network, be it across a local area network (LAN) or over a larger geographic distance, such as the Internet. Each MMCC is associated with a given user, at a given workstation and well-known port. MMCC acts as both a server and client. It notifies the user of connection requests from other users' MMCCs, and places calls to other MMCCs on a user's behalf. Each participant relies on its MMCC to act as an autonomous control entity that decides on the validity of state transitions; it does this by reaching consensus with other potential conferees' MMCCs. It is intended to run continuously as part of the usual workstation environment.

Peer MMCCs orchestrate multisite conferences through use of the Connection Control Protocol (CCP) which handles communication between MMCCs. The control protocol's prime tasks are conference establishment, which includes coordinating resource selection across sites (since we support multiple video codecs), and disconnection. The protocol supports auxiliary tasks as well, such as allowing remote cameras to be switched on request.

The CCP uses UDP [20] to communicate control information between its peers. Since n-way communication is required among conferees, UDP was favored over TCP [21]. Each site only needs one UDP socket [26] to which all other sites can connect, whereas the comparable TCP implementation would require n sockets per site, or n squared sockets total. Furthermore, UDP's connectionless service is ideal for MMCC operation. Since control information is exchanged infrequently by comparison to the duration of a typical conference, TCP connections are considered too much mechanism for the job. UDP does not provide a reliable communication medium, so CCP provides its own reliability mechanisms at the application level. This is an important advantage because in our experience TCP connections do not survive the long duration (hours) of a session due to intermittent Internet network outages.

CCP is essentially transaction-oriented. One MMCC typically sends another MMCC a request, then waits for a reply to determine the next course of action. To support multisite conferences, MMCC often needs to query or submit a request to an entire group of MMCCs at once. CCP currently relies on sequential interaction to simulate multicast. This simplification is made to accommodate limitations in the setup of real-time voice and video streams. A disadvantage, of course, is the lack of parallelism, meaning longer delays before consensus is reached. Nonetheless, MMCC does support some degree of asynchrony by allowing certain local activities to occur while a site is waiting for a reply to its request.

Because of the transaction-oriented nature of MMCC's activities, there was initial consideration of VMTP [4, 5] or Sun RPC [24, 25] as the underlying protocols to use between MMCCs. However, VMTP has not yet been integrated with a released version of the UNIX OS typically running on peoples' workstations, thus limiting the number of places MMCC might run; RPC may become a viable option as the specification becomes more widespread and implementations support the flexibility needed for WAN applications (e.g., flexible timeouts and flexible error handling).

3.1 Connection Establishment and Disconnection

A user submits a list of conferees to MMCC by clicking on participant list entries in a connection pop-up window (see the section on User Interface for further details). That user's MMCC is designated the initiator MMCC. It sends a conference invitation to one remote site at a time. As each conferee is invited, a small pop-up request window appears at the invited site. The keyboard bell tone, the computer equivalent of a telephone ring, also draws attention to the change in state. The invitee is required to answer the request to participate in the conference. As in a telephone call, the initiator will continue to ring the remote user until the request is answered or the initiator clicks a button to give up. If the invitee replies favorably, audio and video connections are established between the initiator and the remote site. In this way, each remote site is invited to join the tele-meeting. If a conferee is not interested in participation (e.g., declines the invitation) or an error occurs (e.g., the remote host is unreachable or a problem arose at the remote site), the initiator asks the local user whether or not the conference setup should continue.

Each remote site is expected to report to the initiator the success or failure of its attempt to set up multimedia connections. The initiator then redistributes this information among all group members by updating each site's record of the participant list and the state information of all conferees. Other sites may join or leave the conference at any time. When a site disconnects itself from a conference, the rest of the conference is normally left intact. However, if the conference was a two-party connection, it is torn down entirely when one of the participants leaves.

3.2 Reliability

To ensure reliability, the connection control protocol uses the usual combination of timeouts and retransmissions. If a remote MMCC does not respond to an initial request after a given timeout period, the requesting MMCC resends the request; the MMCC will retry sending the request some number of times before giving up altogether. The idea is that the timeout duration should be long enough to avoid unnecessary resends which lead to excess traffic, but short enough not to waste time in the event that the original request was actually lost in transit. However, an appropriate timeout length is difficult to predict over such a varied and dynamic community of networks as the Internet. It makes more sense to dynamically adjust timeouts on a site-by-site basis [2, 13, 14], regularly updating the expected roundtrip time for a packet between the two MMCCs, as is done for TCP.

Timeout choice is further complicated by the fact that there are other delays that figure into the complete timeout computation. In other words, a "base" timeout may only be used in those instances where the request-reply sequence is virtually instantaneous. These are cases where the reply is generated either immediately on receipt of a request, or after a very short time. However, there are other cases where request execution time varies. During connection establishment (and in certain other instances) there are activities that prolong a conferee's reply. For example, if a remote site is configured to use a different video codec than the conference initiator site, the remote codec selection needs to be switched to match the originator's. This coordination may take several seconds.

We envision in the near future modifying the protocol to use a two-tier acknowledgment scheme. For requests that require lengthy activity before issuing a response, the recipient of the request will send an immediate acknowledgment (ACK) of receipt of the request. The recipient will follow the ACK with a full-fledged REPLY once the result of the request is learned. Granted, this means that a timeout period will need to be set for an acceptable inter-ACK-REPLY interval. That interval may vary depending on the type of request made of the remote site. The ACK message is designed to contain a suggested inter-ACK-REPLY timeout.

As timeouts are based on the activity at hand, so are the protocol reliability requirements. Actually, because the functions MMCC supports range from crucial to peripheral, its policy on reliable communication varies depending on both what the request is and when the request occurs. It is most strict about reliability during conference initiation; two sites *must* hand-shake correctly during the connection procedure, otherwise the conference is never established. Reliability is still important, but somewhat less crucial at disconnection time; a site tries its best to inform a remote site that it is leaving the conference, but if no reply is heard from the remote site and all retries have been used, it leaves the conference anyway. MMCC is least strict about reliability once the conference is in place. For instance, if a request to switch a remote site's cameras goes unanswered, the local site is informed of the connectivity problem but the conference remains intact until the local site disconnects.

Most of the responsibility for reliability lies with the MMCC that initiates a command request. If a response takes too long, the request is resent and MMCC retries a reasonable number of times before giving up. Typically, the responding MMCC need only resend the reply on receipt of a repeat request. However, during connection establishment, a responding MMCC has to be able to recognize when the initiator MMCC has for some reason gone away leaving the responding MMCC in a transitional state, neither connected nor idle. This may happen, for instance, when the network becomes partitioned, so communication is interrupted between the two sites. Therefore, during connection establishment, the responding MMCC also uses timeouts and retransmission to detect when the requester is no longer reachable.

3.3 Re-synchronization

Due to the distributed nature of the system, cooperating MMCCs may sometimes get out of synchronization. Site A may think it is in conference with site B, but site B may think it is not in conference at all. This may be the result of an unsuccessful disconnect, due to network partitioning, or a user unexpectedly restarting the MMCC program while it is connected. If site A subsequently receives a conference request from site B, it realizes something is amiss and attempts to resynchronize its state with site B. To try to ascertain its real state, site A also communicates with any other sites with which it still thinks it is in conference, potentially correcting other sites' state information in the process.

MMCC also tries to detect and resolve connection collisions. These are instances where connection requests between a pair of sites cross in transit, i.e., site A sends a request to connect to site B, at the same time B sends a request to connect to site A. It currently performs a comparison of network addresses; the site with the greater address is selected to continue as the initiator and the other site goes back to idle to let the connection complete as an invitee.

4. Voice and Video

Before MMCC existed, there was the Voice Terminal program (VT), and the Packet Video Program (PVP). They were used as separate programs to establish audio and video connections respectively. This was done through command line interfaces. One of the early goals of MMCC was to hide the details of setting up voice and video connections through these system components and instead use MMCC as a front end to one VT and PVP pair.

The VT and PVP programs were originally implemented on a BBN Butterfly multiprocessor (see Figure 2 below). MMCC is usually run from a workstation in the same room as the audio and video equipment, so it communicates with the pair across the network. VT and PVP were intended to be co-located with MMCC (on the same LAN), but there is nothing to prevent greater separation: for instance, conference operators have been known to run MMCCs that control VT/PVP pairs located across the country or located on another continent.

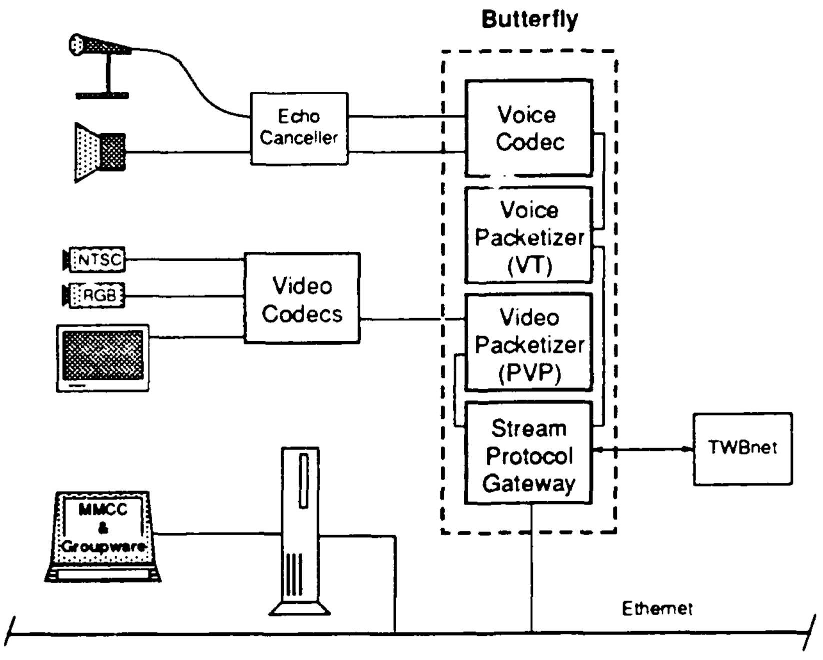


Figure 2. System Components

Current versions of VT and PVP have been implemented for the SPARCstation platform and may be ported to other workstations in the future. There is also no reason why video, voice, groupware and MMCC processing couldn't all reside on the same workstation provided the architecture of the machine (CPU, OS, network interface et al) could meet the performance demands.

VT and PVP digitize and packetize data, using the Network Voice Protocol (NVP) for audio [6] and the Packet Video Protocol (PVP) for video [7]. They transmit this data across the network using the experimental Stream protocol (ST) [28] and the TWBnet, both of which provide performance guarantees to the real-time data through use of resource reservation. A diagram of the communication protocol stack is displayed in Figure 3.

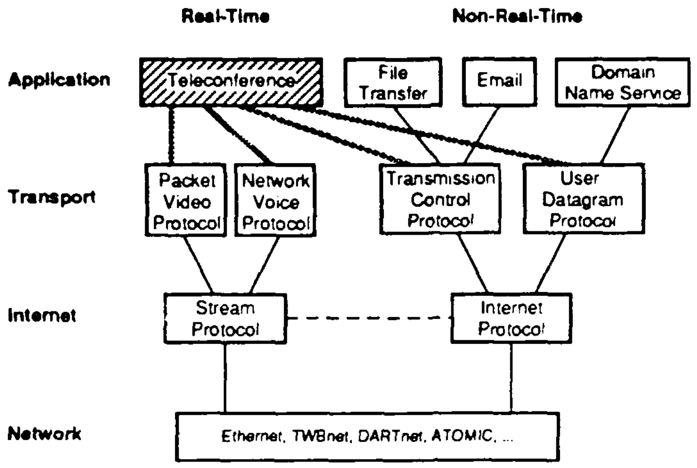


Figure 3. Layered Protocol Stack

As the Initiator's MMCC receives approval from each peer MMCC for the connection, voice and video connections are set up via their VTs and PVPs. Originally, MMCC communicated solely with VT to set up these connections (see Figure below). VT in turn communicated conference state to PVP via shared memory. This chain of command is largely historical and has the drawback that MMCC is not at liberty to set up strictly voice or strictly video conferences, only joint voice and video ones. MMCC sends connection requests to its local VT using NVP, the VT-to-VT protocol. VT receives and acts on these requests, but is unable to report feedback to MMCC on whether or not the voice and video connections succeeded (this is due to a Butterfly VT implementation that is not complete).

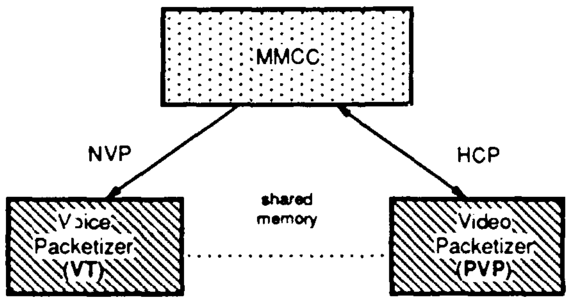


Figure 4. Communication with VT and PVP

More recently, we have incorporated the Host Control Protocol (HCP) [17] in PVP as a control channel for three purposes:

1. To establish video conferences with in-band audio through communication directly with PVP. In these instances, no separate voice packetizer is used. On arrival, the combined data stream is processed by PVP as usual, then forwarded to and unbundled by the video codec. HCP is able to report explicit connection results when the connection is established through PVP in this manner.

2. To learn the outcome of connection requests that were established through VT.

3. To support video stream selection in multisite conferences. Although PVP receives video streams from all participating sites, depending on the type of video codec in use it only delivers to the codec between one and four streams at a time. HCP may be used between MMCC and PVP to select among and to organize the display of the video streams.

As we move to new implementations of the real-time programs for the SPARCstation platform, we plan to ommunicate with each module independently via HCP, or possibly SNMP [30] or RPC. This eliminates the need ır shared memory between VT and PVP. This also means the details of connection failures will no longer be mbiguous, that the system will have a more complete monitoring capability, and that separate audio- or ideo-only conferences could be established. There are also plans to use such a control channel with VT for udio muting and possibly for volume control of individual audio streams.

. Configuration Management

From MMCC's point of view there are two configurability issues; addressing of remote teleconferencing sites nd details regarding how each site is configured with multimedia devices.

.1 Addressing Information

MMCC obtains some of its configuration information from a local file that is read at start up. The purpose of ıe configuration file is to hide addressing details from the user. It associates an alias with each peer MMCC. hese aliases are used in the display of participant lists. It provides the Internet address and port of all peer IMCCs. Although there is a default MMCC port number, one can use a different port and run multiple MMCCs om the same machine (for the DWS network, BBN controls four MMCCs from one workstation, where each IMCC corresponds to a distinct VT/PVP pair). In addition to peer MMCC addresses, the configuration file stores T and PVP addresses.

.2 Device Coordination

Another aspect of configuration management is coordinating cameras, monitors, codecs, audio equipment, nd other gadgets at each site. Typically, each site is configured with a room view camera that is mounted above ıe video monitor and displays the participants, and a copy stand camera for slides or graphic stills.

Each MMCC keeps track of the different numbers and types of hardware devices available at its own site and ses this information to decide if connection requests can or cannot be met. However, these configuration ifferences are not the only form of heterogeneity in the system. Each site can select among several viable onfiguration alternatives every time a conference is set up. Notably, most sites have two different video codecs. hese codecs support several different modes and a range of data rates. It becomes MMCC's task to make sure ıat these configuration options are coordinated when a conference is established; MMCC makes sure the codec ʼpes and video data rates match at all sites and switches them if they do not.

.3 RPC-based Media Services

To coordinate the multimedia devices, each MMCC relies on several servers that are accessed using Sun PC. The servers communicate with these devices over RS232 serial lines. Each server is a separate program ıat either is co-located with MMCC on the same workstation or executes on another workstation across the local thernet. So far, there are three such servers (see Figure 5, MMCC Server Associations). They attempt to mask ıe hardware specifics of the devices they control. MMCC does not require that each site necessarily be onfigured with all servers for MMCC to function.

One reason for making a separate server out of the routines that control a device is for modularization. nother reason is location-independence: to be able to attach the device to any machine, preferably one onveniently located near the device. A third reason is that there are are a limited number of serial ports per orkstation (e.g., only two per Sun). With more devices than serial ports, some devices must be attached to ther machines anyway.

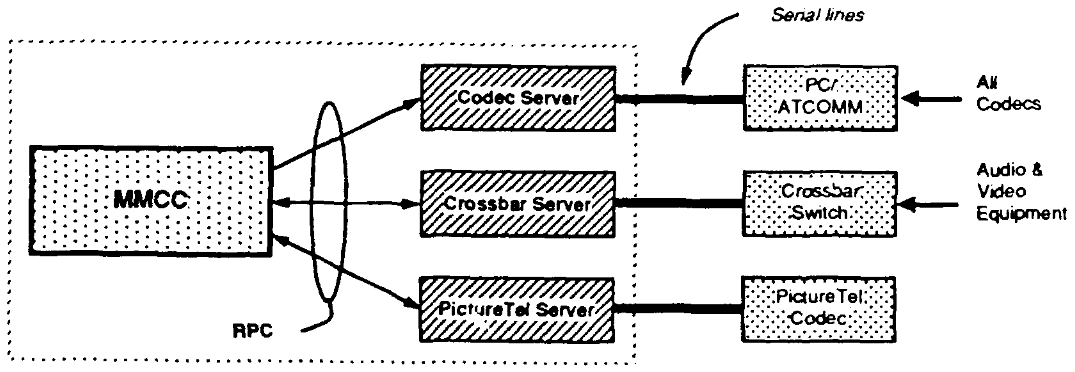


Figure 5. MMCC Server Associations

5.4 Server Functionality

The codec server is primarily responsible for video codec and monitor control, as well as providing a fair amount of debugging support. It communicates over the serial port to a PC plus an ATCOMM board to which all codecs connect.

The crossbar server is a switching facility that allows selection of various audio and video configurations. It supports combinations of multiple cameras, monitors, codec types and a variety of audio equipment. There are also several sites that are multi-room facilities which rely on the crossbar to switch among rooms. The crossbar server attaches to the crossbar switch, which in turn is connected to most everything.

Finally, the PictureTel server is specifically tailored to control the PictureTel codec to support multisite conferences. The multimedia conferencing system sends video streams from all sites to all other sites, but uses the video packetizer program to decide how many streams of data to forward on to the codec in use. In the PictureTel case, PVP forwards one video stream since the PictureTel is only capable of displaying one video image at a time. During a multisite PictureTel conference, MMCC provides either receiver-selected or sender-selected floor control. Receiver-selected floor control allows each site to choose whichever remote site it prefers to watch. In contrast, sender-selected floor control allows only one site at a time to actually send its video to all other sites; accompanying audio, however, may be sent separately from all sites to all other sites. This mode of operation is crucial to support limited bandwidth scenarios, where the bandwidth needed for all sites to send information to all other sites exceeds upper limits. MMCC supports both floor switching modes by delivering control commands to the PictureTel server to switch between point-to-point and multipoint mode, and to refresh/resynchronize the local image as needed. MMCC also tells PVP which video stream the user is interested in watching, and, in limited bandwidth cases, whether or not to send video to the remote sites. The PictureTel server attaches to the control-a port of the PictureTel codec.

Note the degree to which we can both monitor and communicate with these services. The codec server, like VT, only supports the unidirectional flow of information. There is no return channel to learn whether or not a request for service has succeeded. In contrast, the other two servers allow bidirectional communication between the servers and the devices, so errors may be reported locally and/or remotely.

6. Integration with Groupware

MMCC interacts with the BBN MMConf program [8] to support a suite of shared applications, ranging from shared documents, to shared presentation tools, to shared video map browsers, et cetera. MMCC can automatically establish and disconnect an MMConf session in parallel with voice and video connections, bringing up MMConf in a designated conference directory. In addition, MMCC interacts with Mbftptool, a window-based application to run multiple background FTPs among conference participants [10]. Mbftptool may be used both before and during conferences to prestage transfers of files intended to be shared.

MMCC uses a conference identifier to loosely couple with these conferencing tools. The conference ID is usually the subject of the meeting or the name of the group that is gathering. The conference identifier is assigned on a per conference basis and is used as the name of the "conference directory" to which shared applications connect and in which shared files are placed. The conference ID is distributed to participating sites by MMCC, which also makes sure the conference directory exists at each site. The conference ID may be set at connection initiation or beforehand, in anticipation of an upcoming conference.

MMCC, MMConf and Mbftptool are independent applications that support a conference mode. Integration with MMCC is possible because MMConf and Mbftptool allow other processes to rendezvous with them via some form of interprocess communication. MMCC forks these applications and is able, through a combination of signals, control ports, process identifiers, files and command line arguments, to convey state information and state changes to them.

MMCC has no need for a password during connection attempts; authorization comes from the person who runs MMCC after logging in. However, MMConf and Mbftptool, like other conferencing applications, often include a password verification step, since they allow one user to run the application for another user.

The interaction between MMCC and these groupware tools has raised two questions:

1. How can we avoid redundant steps needed by all conference-oriented software during conference setup?
2. How can we provide a more general way for groupware to be integrated into the conferencing system?

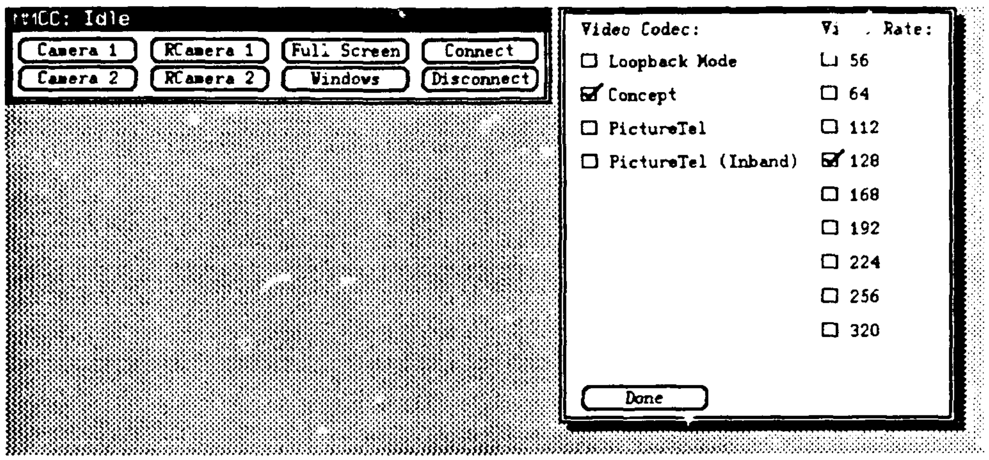
We believe there is utility in providing a conference server that handles "the conference" and attaches or detaches services accordingly. We envision it as a connection management layer that resides below the user interface, but above all media (voice, video and groupware). This layer would have clearly-defined interfaces with the layers above and below to allow multiple implementations of functionally equivalent or near-equivalent services. Among other things, the conference manager would avoid duplication of effort in conference-oriented applications; this encompasses management of participation (state information, conferee lists), authentication (passwords), and presentation of coordinated user interfaces. This is along the lines of Swinehart's switching kernel [27] and in some regards is related to Lantz' and Lauwers' models for user interfaces that endow programs with the smarts to be multi-user [16]. This idea is discussed further in the section Future Directions.

7. User Interface

MMCC usually appears in a corner of the computer screen. It's intended to be "conspicuously inconspicuous"; out of the way, but immediately available. The information and functions most often used are those items most evident: status information, local and remote camera control, monitor layout, and connection and disconnection buttons.

Although live motion video is currently displayed on a video monitor that is separate from the workstation monitor where MMCC runs, we are ultimately interested in displaying video-in-a-window on each teleconferencing workstation. Ideally, we'd like to display video for each remote site in its own window on the workstation screen. In the present system, monitor layout options vary depending on the video codec selected (and on how many simultaneous video streams a codec can accept). The monitor can be used to display video streams from up to four sites in separate quadrants or to display a single image using the full screen.

Other hidden functions appear as options in MMCC's frame menu -- the pop-up menu that appears when mousing on the edge of the MMCC window frame. Some of the more interesting hidden selections include Autopilot control (a discussion of which follows in the next section, Current Use) and Codec control. Figure 6 below shows a typical codec pop-up panel. In this case, MMCC is not in conference and the Concept video codec is selected to operate at 128 Kbps. When MMCC is run in debug mode, other buttons appear on this panel including ones to print and zero codec counters.



Video Codec:
☐ Loopback Mode
[✓ Concept]
☐ PictureTel
☐ PictureTel (Inband)

Vi . Rate:
☐ 56
☐ 64
☐ 112
[✓ 128]
☐ 168
☐ 192
☐ 224
☐ 256
☐ 320

Figure 6. Codec pop-up window

MMCC is generally run as a window-based program, but it may also be run in a command-line mode. This enables running the program through a Telnet connection to a remote site where one does not have access to the remote window manager. The command-line mode of MMCC, mostly used fo: debugging purposes, only supports client functions (receiving connection requests but not initiating them).

8. Current Use

The ISI/BBN teleconferencing system currently provides meeting-room-to-meeting-room style conferences. A few people convene per site, then a conference is established among the different sites. A workstation resides at each site where a general purpose teleconferencing account is used to run a copy of MMCC. This setup is a result of practicality -- equipment costs -- rather than intention. Those who use the system borrow the conference room, the teleconferencing account and workstation, and typically have no interest in conference establishment. They want to arrive and have the conference already in place.

8.1 Policy on Remote Control

To facilitate this mode of operation, we added what we refer to as Autopilot mode. A remote site can be instructed to automatically answer yes to connection requests. This is helpful when MMCC is already running at a remote site and no one is around to click on the right buttons for confirmation. On disconnection, any site connected to in this fashion will be automatically disconnected. Or, the controlling site may selectively disconnect Autopiloted sites and leave the rest of the conference intact. Autopilot mode generally lasts for the duration of a conference. Autopilot mode is a valuable asset in the current system because it allows remote control of remote sites, and also facilitates remote repair. After a network partition, a remote site may remotely disconnect a site that is out of synchronization.

Admittedly, this "trap door" does pose a security risk. We minimize this risk in the current system because we are able to make assumptions about who is using the system. MMCC only runs at the teleconferencing sites, only under a teleconferencing account, and is configured to a restricted group of participants. However, the Autopilot capability must be re-thought as we move to personal conferencing, where a collection of varied users, with varied configurations will manage their own conferences. Additionally, MMCC will need to extend its model of the system from being site-specific to user-specific, since it currently considers an endpoint of a conference to be a host address and port, rather than a tuple that includes a user ID.

Autopiloting definitely raises the issue of remote control: to what degree should remote sites be able to control a local site's resources (e.g., cameras)? Because the answer inevitably varies -- depending on everything from each participant's feelings on the matter, to the formality or informality of the meeting -- MMCC provides a range of exclusivity levels. A site may decide it does not want remote sites tinkering with its local cameras; remote camera control may therefore be disallowed. A site may also establish an exclusive conference, in which case any incoming requests for participation are deflected without troubling the already-in-conference members. An option is also in the works for a site to disallow other sites from using Autopilot to set up connections with it.

From an operations point of view, the continued availability of an override mechanism is attractive. With the current system, situations have arisen where a remote operator either would like to or needs to "fix" an otherwise "broken" site. The current strategy is that MMCC leaves exclusivity levels wide open by default.

8.1 Optimizations for Conference Establishment

The system's current use has also had an impact on the conference establishment protocol. Generic conference establishment might include three phases:

1. Find out if those invited to conference are interested in participation.
2. Orchestrate whichever media connections are requested.
3. Report the status of the connection attempts to all those involved.

Because MMCC is used in a more formal setting, conferences are actually pre-arranged to begin at a given hour, among an agreed-upon group of conferees. This renders the first phase of connection establishment less meaningful than in an informal conference where the conference initiator is really trying to determine if it is ok to set up a meeting.

Notice Figure 7, Multisite Conference Establishment, below. In our system, for all but the first pair of conferees, the first two phases are combined -- not only because the connection request phase is of a secondary nature, but also because of the potential for large delays in a WAN setting and because of the Caller/Callee connection order requirements of the VT program. For all invitees beyond the first one, a favorable reply is returned with the outcome of the connection attempt. Note also that the initiator establishes audio and video connections with the first invitee (step 3 below), whereas subsequent invitees initiate audio and video connections to the initiator to join in (step 5 below).

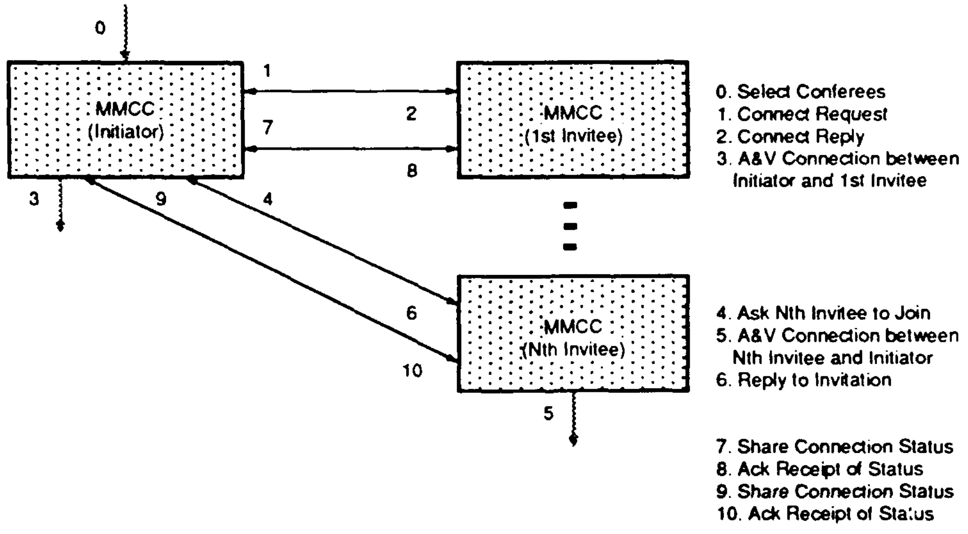

0
MMCC (Initiator)
MMCC (1st Invitee)
MMCC (Nth Invitee)
1
2
7
8
3
9
4
6
10
5
0. Select Conferees
1. Connect Request
2. Connect Reply
3. A&V Connection between Initiator and 1st Invitee
4. Ask Nth Invitee to Join
5. A&V Connection between Nth Invitee and Initiator
6. Reply to Invitation
7. Share Connection Status
8. Ack Receipt of Status
9. Share Connection Status
10. Ack Receipt of Status

Figure 7. Multisite Connection Establishment

The multisite conference establishment procedure has been optimized for this operating environment in which call requests are rarely rejected. However, there are other scenarios where it would make more sense to wait until all participants have agreed to conference before establishing media connections that may end up having to be torn down anyway (e.g., because a critical group member is unavailable to meet). In these scenarios, users may want to have control over the phases of the connection establishment procedure.

9. Feasibility of the Model

There are several difficult aspects to providing a robust and rich conference control service. The difficulties mainly lie in running a distributed program across the Internet, but there are other concerns: dissimilar routing for the different media, providing complete monitoring information, handling system errors intelligently, and the growing heterogeneity among site configurations and system scalability.

9.1 Distributed Program Complications

MMCC suffers from the usual distributed program complications. There is no central server for definitive state information, so MMCC needs to withstand network partitions and broken or transient routes. All of these lead to curious behavior in distributed programs that rely on n-way connectivity. Because MMCC's control protocol is based on UDP, it is less disturbed by network breaks. However, shared applications such as MMConf that use TCP for reliable delivery and synchronization of shared data files are more sensitive to network outages. They usually have no provisions for repairing a group session in the face of lost connectivity of a group member. MMCC is somewhat better in that it does not tear down the connection until disconnection time. MMCC also institutes a re-synchronization step when it finds a discrepancy in site state information. But, MMCC only repairs state information on discovering it is incorrect, rather than checking regularly.

9.2 Heterogeneity and Scalability

If our experience in teleconferencing is any indication of developments to come, heterogeneity among conference sites will continue to increase as the system scales up. These variations require flexible configuration management.

MMCC must ensure that when a request for a media device is made, the device exists, is available and its requirements (e.g., bandwidth or coding algorithm) can be met. A standard configuration language is needed to describe sites' capabilities, so that MMCC may negotiate a common configuration between conference sites during connection set up. As the community of teleconferencing sites grows larger, there is also incentive to move away from replicated configuration information and move instead toward obtaining site-specific information on demand.

The same holds for addressing information. As more user's have the ability to conference, it will not be feasible for MMCC to store all participants' addressing information in its local database. Therefore, address information will need to be provided more dynamically to the system, through the use of a conference-related directory service. The address for each user's conferencing workstation (MMCC) will be obtainable through the directory service, but again more detailed information, such as VT and PVP addresses or addresses for other system components, will only be shared on demand to cut down on the overhead expected with large numbers of teleconferencing users.

There is also motivation to introduce more parallelism in the control protocol for scalability sake. For a sizeable conference, sequential communication with each user becomes too time-consuming for approximating group interactions.

9.3 Routing of Media

In the MMC testbed, real-time data specifically travels over the TWBnet, since it supports bandwidth reservation. However, control information and shared workspace data may travel over any number of Internet routes, including, but not exclusively the TWBnet. At times, this leads to some media appearing to be available while other media are not; peer MMCCs can communicate, but peer VTs or peer PVPs cannot, or vice versa. With or without the restricted access of real-time data to the TWBnet, the requirements for real-time data are different enough from non-real-time data that the routes may be different anyway; this will result in the same phenomenon.

Carrying separate media over independent data paths means they may get out synchronization. This is true even between the real-time audio and video streams. So far, this has not been a problem largely because the voice and video data traverse the same experimental network, and consequently their delay characteristics are nearly identical. The fact that they are not exactly synchronized is not usually noticeable since the resolution of the video, in combination with the typical distance of the user(s) from the monitor, makes it difficult to see precise lip motion. However, for future scenarios which might need more accurate synchronization, the Multi-Service Flow Synchronization Protocol is under consideration [12]; the approach underway is to integrate synchronization mechanisms in multiple levels in the system (i.e., MMCC, VT and PVP).

9.4 Monitoring and Error Reporting

MMCC is the upper-most layer in a series of layers of interface to the conferencing system; it relies on numerous hardware and software components. MMCC can easily track the status of peer MMCCs and less easily track reliability of VT or PVP hosts, but cannot predict when the TWBnet, or a leased-line or a gateway or a modem is impaired for the day. Our experience has been that, of the numerous devices used in a conference, only some of these can report status information. Better monitoring is crucial for personal conferencing to be trusted as a reliable form of collaboration.

Error reporting presents other difficulties. MMCC has had to characterize operation failures into error classes. Three typical classifications are whether the error was local, occurred at a remote site, or was the consequence of Autopilot activities. Each case effects the way errors are reported: if the error occurred locally, then the error is displayed in a pop-up window; if the error was generated remotely, information about which remote site triggered the error must be included as well, and the error is reported simultaneously at the remote site; if the error occurred while the site was being Autopiloted, the error must not be displayed in a pop-up, but rather should be printed on to the standard error output, since there is no way to ensure that an individual is present at the console to view or remove the error panel. To make error reporting more precise, all errors are timestamped with the date and time; this avoids having an old error mistaken as a new error, since a site's teleconferencing system and meeting room are shared among all local users. Errors are also categorized according to severity. They either cause a connection to abort or may simply result in a warning. For example, if codec synchronization fails during conference setup, then the conference is aborted. However if a site is unable to switch a camera once in conference, the system considers it a non-critical error and the conference proceeds anyway.

10. Future Directions

MMCC embodies many different conference management tasks. The evolution from simply managing local camera and monitor control, to incorporating support for point-to-point conferences, to providing multisite and multi-configured teleconferences, has unearthed several shortcomings in its original design. This section coalesces some themes for an improved conference control architecture.

10.1 Personal Conferencing: Inter-office telecollaboration

To complement formal meeting facilities, MMCC needs to support highly informal, impromptu conferences where individuals rendezvous as desired [22]. Therefore, the project is moving away from inter-meeting-room conferences and toward personal conferencing in the form of inter-office collaborations. Although office-to-office conferencing may arise on a more frequent basis, there is still a place for meeting room conferences, for example when many people need to conference simultaneously. The connection management architecture should support both these modes. There is nothing inherently different about them except that an inter-office setting has the potential to quickly scale to be used by many more individuals and to demand a richer interface (e.g., personalized call filtering functions). An inter-meeting-room system is likely to warrant special privileges such as Autopilot for remote operators to oversee the system.

There are also scenarios that have not arisen in the current setting, but need to be supported in the inter-office setting. An example is multiple simultaneous conferences per user. In the personal conferencing domain, these might take the form either of sub-conferences (some subset of participants) or separate but potentially overlapping conferences that use any combination of media with any combination of participants. This capability has not been addressed in our system. In addition, ideas such as call forwarding and visiting, which have been successful in LAN settings [27], need to be explored as useful tools in wider area environments.

10.2 Conference Manager Abstraction

Perhaps the most important conclusions drawn from our experience with conference control are the need for a conference manager abstraction, and the need to provide well-defined interfaces between system components. Compare the earlier system diagram with Figure 8, Conference Manager.

The current train of thought is to distill MMCC into primarily a connection manager. It would have well-specified interfaces to the user above and to the separate media below. The idea is to support multiple user interfaces (i.e., local or remote, independent or overlapping in functionality), as well as multiple media-specific (audio, video, groupware) agents below. These media-specific agents would inherit the responsibility of knowing the details of communicating with the RPC-based media servers. An agent would only know about the servers which controlled devices in the agent's media category. In some cases, as for the crossbar switch server, a server would be known by more than one type of media agent. In general though, the goal would be to localize information. Agents managing the same media type might be swapped for each other. For example, an agent that dialed a telephone might substitute for one that placed a packet audio call. The difference between these agents would be that they communicate with different underlying services and communicate differently with their peer agents across the network.

Without question, the present media servers we have developed are specific to the particular equipment selected for the MMC system. They could be considered the forerunners to a whole collection of media servers that might populate each site. A future direction is to explore a more unified model for these services, in other words, to establish well-defined interfaces between the media agents and the servers to allow separate but similar devices (e.g., different video codecs) to substitute for one another (e.g., when the requested service was busy or unavailable) [19], in the same way agents might substitute for one another. This task includes the creation of a description language to characterize the capabilities of devices and available resources. This language would be used by media agents to choose among underlying options.

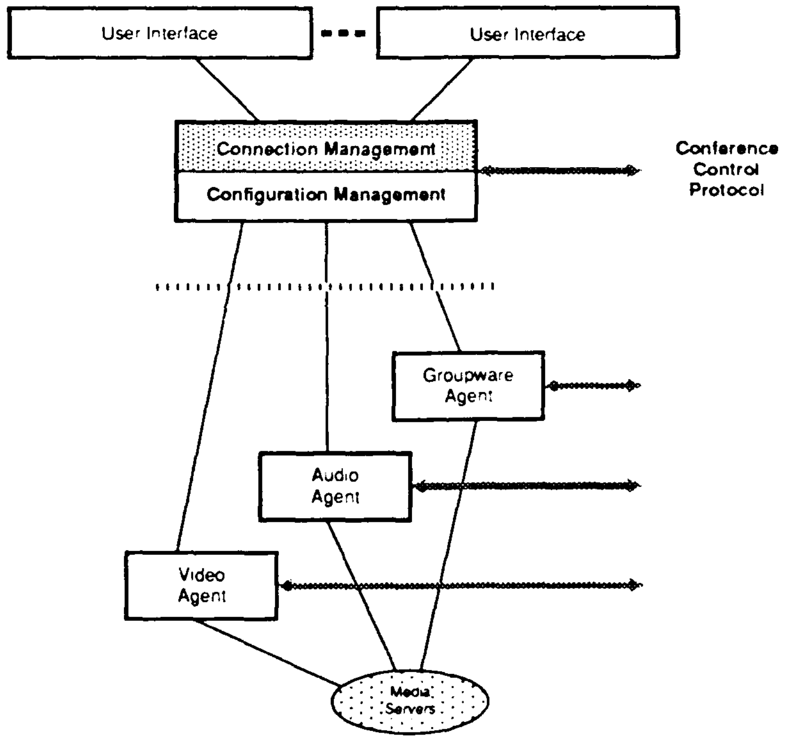


Figure 8. Conference Manager

In many ways configuration management remains the same, but direct interaction with the media servers would be relegated to the media agents. Configuration information would be shared with MMCC through the media agents and would allow user interfaces to be fashioned to match a site's particular configuration.

Separate media handling means conference negotiation could be richer. During conference setup, the system should be able to make a best-effort attempt to meet the request requirements. In situations where the requirements cannot be met (e.g., one or some of the sites don't support video, but only audio, or if only a limited amount of bandwidth is available so the video quality must be slightly compromised), the system should have enough heuristics built in to meet the request as best as it can.

All components in the system need not be co-located with one another. The intent is to loosely couple them via RPC or another light-weight transaction-oriented protocol.

For communication with remote sites to work, the application, transport and internet protocols need to match between peer components at all sites. No particular protocol need be required, so media agents can select the protocol best suited to their needs. Discussion is underway to create a specification for a connection control protocol that could interoperate with other conference manager implementations [1, 18, 19, 29].

The architecture also needs to support an interface for groupware tool builders. Shared applications should be able to link into the system easily (e.g., gain access to participant lists and state information).

10.3 Extensions to the Connection Control Protocol

There is the ever-present goal to provide an efficient (with minimal delays) and robust conferencing service in a WAN environment. A number of modifications are planned for the connection control protocoi to meet these goals. They include extensions to the timeout scheme, consideration of parallel and multicast communication, and the incorporation of additional resynchronization mechanisms.

Timeouts. First, timeouts must be made variable. They should be based on dynamically updated roundtrip times. The timeout period will therefore be stored on a per site basis. This should better balance the need to quickly resend lost packets (to reduce delay) and the desire to avoid overloading the network with unnecessary resends (when the resend time period is too short). Next, a two-tier ACK/REPLY strategy should be employed. For those activities that will take longer than the allotted timeout period, the protocol should be able to send an acknowledgment of receipt, with a real reply being sent later when the task has been completed. The ACK will contain an approximation for how long the task will take, so MMCC can timeout accordingly. This allows the protocol to flexibly support a range of activities and allows each site to set its own guidelines for how long those activities will take (sites may be configured differently, resulting in longer/shorter times to perform a similar task).

Parallel and Multicast Communication. MMCC is a transaction-based entity. It currently uses sequential communication with a group of peers. In other words, it communicates with one peer at a time to reach group decisions or to complete group updates. Changes to the timeout scheme to support multiple queues and improvements to the state machine will allow MMCC to provide for more parallelism. Instead of waiting for a reply from one peer before moving on to the next, MMCC would initiate all requests simultaneously. The control protocol will need to experiment with efficient ways to handle the added asynchrony. Furthermore, use of IP multicast [9] may be in order as well.

Resynchronization. The present approach is admittedly quite passive. MMCC does not set out to locate mismatched states among peers. Instead, it waits to discover such problems through interactions like connection or disconnection requests (when a remote site requests a connection, but the local site thinks it is already in conference with that site). In the future, MMCC might reduce state mismatches by including conference state information (the local and remote participants' states) with every message exchanged. In addition, each MMCC might also specifically check the conference status by probing the remote conferees at regular intervals. Other ideas from distributed databases might also be incorporated.

10.4 Improved Communication with VT and PVP

Much of the complexities as well as deficiencies of setting up voice and video connections are due to the lack of a well-defined interface between MMCC and the VT and PVP programs. A great improvement to the system will be the use of a single protocol to exchange information between the real-time processing programs as implemented on the SPARCstation and MMCC. Whether that protocol is HCP or SMTP or RPC, the resulting interface will make the system more robust. MMCC will be able to collect detailed feedback from VT and PVP. In the new scheme of things, VT and PVP would be considered media agents and would have richer functional interfaces than at present. This means that separate control of audio and video would be possible, so that conferences of just audio or just video could be established. It also means that audio functions could be added (e.g., muting, balancing, individual stream volume control). In addition, MMCC would specify codec and data rate selection to VT and PVP, which in turn would take on the responsibility of making those selections by interacting with the requisite media servers.

10.5 Configuration File Redesign

The current MMCC configuration file contents are to be separated into two different files: an address file for remote user information (e.g., remote host aliases and addresses), and a configuration file detailing local system attributes (e.g., how many cameras are available, and the types of video codecs supported). Dynamic modifications to this information must be allowed, whether this means re-reading the files or allowing users to enter new or modified information on the fly.

An MMCC would share local site-specific information with remote MMCCs on demand, say during the connection establishment phase or when the information is relevant (e.g., when a site wants to be able to switch remote cameras and needs to know how many remote cameras are really available). This would eliminate the need for all MMCC configuration files to be identical (the configuration file is currently replicated at each site). It also means that when an entry in the configuration file changes, the update would only have to occur at one location -- the site that "owns" the change. This not only scales better, but also lets MMCC make the more realistic assumption that sites are NOT identical. What one site knows or needs to know about remote sites does not have to be shared universally.

During this redesign, it also makes sense to allow user preferences to be specified. These might range from a default system configuration (e.g., which codec to use by default), to finer access control that could be used for call filtering or call prioritization (who may or may not join or interrupt a conversation), to more specific text for buttons and pop-ups for the user interface.

10.6 Miscellaneous

Near term plans include porting MMCC to run under an X environment to allow use of other workstation platforms. One issue for investigation is the portability of Sun RPC which is used between MMCC and the servers.

Automation of the conference setup procedure is under consideration; the conference manager might automatically remind individuals beforehand of an upcoming tele-meeting and might also take it upon itself to automatically initiate the conference at the scheduled hour. The scheduler might also pass resource reservations to the network (a la ST) to ensure that scheduled meetings will actually be able to take place when agreed upon.

Teleconferencing represents only one of the many possible multimedia applications. It would be beneficial for the MMC system to couple with other multimedia facilities, such as recording and playback of voice and video, text-to-speech synthesizers, radio or TV broadcast services, voice mail, and the telephone. We are laying the groundwork for an architecture to accommodate these modes of operation.

11. Conclusion

During its tenure as a prototype interface to the ISI/BBN multimedia conferencing system, MMCC has provided conference orchestration, local and remote media control, and coordination with groupware applications. Its distributed, peer-to-peer architecture and application-level connection control protocol have offered insights into what's difficult about making conference control work across the Internet: variations in network and system delays, communication outages, state resynchronization, and heterogeneous site configurations. In short, MMCC has highlighted several important requirements for distributed programs to operate effectively across wide area networks. For personal conferencing to gain wider acceptance as a convenient and reliable form of collaboration and to become what some predict will be the *E-mail of the 90's*, these difficulties must be overcome.

Additionally, MMCC has contributed toward our understanding of the functional interfaces needed between system components and of the modularity needed by the system in general. Its design has raised issues about robust services, scalability, and interoperability with other conference manager implementations. Finally, MMCC has been a springboard for ideas about a conference layer abstraction that has the potential to provide a fully integrated multimedia conference control service. The blueprints for future research in this area have been outlined in this paper and we are now turning our efforts toward the implementation of these goals.

Acknowledgment

Many thanks to Steve Casner, Deborah Estrin and Jon Postel for their extensive comments on earlier drafts of this paper. This research was sponsored by the Defense Advanced Research Projects Agency under Contract Number DABT63-91-C-0001. The views and conclusions in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency of the U.S. Government.

References

[1] Bates, P.C., Segal, M.E., "Touring Machine: A Video Telecommunications Software Testbed", *Proceedings First International Workshop on Network and Operating System Support for Digital Audio and Video*, Berkeley, CA (Nov 1990).

[2] Braden, R. (ed.), "Requirements for Internet Hosts – Communication Layers", RFC 1122, USC/Information Sciences Institute (Oct 1989).

[3] Casner, S., Seo, K., Edmond, W., Topolcic, C., "N-Way Conferencing with Packet Video", *Proceedings 3rd International Workshop on Packet Video*, VISICOM '90, Morristown, NJ (Mar 1990).

[4] Cheriton, D., Williamson, C., "VMTP as the Transport Layer for High-Performance Distributed Systems", *IEEE Communications Magazine*, pp.37-44 (June 1989).

[5] Cheriton, D., "VMTP: Versatile Message Transaction Protocol - Protocol Specification", RFC 1045, Stanford University, Stanford, CA (Feb 1988)

[6] Cohen, D., "Specifications for the Network Voice Protocol, Technical Report ISI/RR-75-39, USC/Information Sciences Institute, Marina del Rey, CA (Mar 1976).

[7] Cole, R., "PVP - A Packet Video Protocol", Internal Document, USC/Information Sciences Institute, Marina del Rey, CA (July 1981).

[8] Crowley, T., Forsdick, H., Milazzo, P., Tomlinson, R., Baker, E., "Research in Real-time Multimedia Communications Applications", Report No 7325, BBN, Cambridge, MA (May 1990).

[9] Deering, S., "Host Extensions for IP Multicasting", RFC 1054 (May 1988).

[10] DeSchon, A., Braden, R., "Background File Transfer Program BFTP", RFC 1068, USC/Information Sciences Institute, Marina del Rey, CA.

[11] Digital Multimedia Systems, *Communications of the ACM*, Vol.34, No.4 (April 1991).

[12] Escobar, J., Deutsch, D., Partridge, C., "A Multi-Service Flow Synchronization Protocol", BBN Systems and Technologies Division, Cambridge, MA (Mar 1991).

[13] Jacobson, V., "Congestion Avoidance and Control", *Proceedings ACM SIGCOMM '88* (Aug 1988)

[14] Karn, P., Partridge, C., "Round Trip Time Estimation", *Proceedings ACM SIGCOMM '87* (Aug 1987).

[15] Kraut, R., Egido, C., "Patterns of Contact and Communication in Scientific Research Collaboration", *Proceedings Conference on Computer-Supported Cooperative Work*, Portland, OR, pp.1-12 (Sept 1988).

[16] Lauwers, J.C., Lantz, K.A., "Collaboration Awareness in Support of Collaboration Transparency: Requirements for the Next Generation of Shared Window Systems", *Proceedings CHI '90 Conference on Human Factors in Computer Systems* (1990)

[17] Leib, M., "Host Control Protocol", BBN, Cambridge, MA (Apr 1990).

[18] Leung, W.H., Baumgartner, T.J., Hwang, Y.H., Morgan, M.J., Tu, S.C., "A Software Architecture for Workstations Supporting Multimedia Conferencing in Packet Switching Networks", *IEEE Journal on Selected Areas in Communications*, Vol 8, No 3, pp.380-390 (Apr 1990).

[19] Nicolaou, C., "An Architecture for Real-Time Multimedia Communications Systems", *IEEE Journal on Selected Areas in Communications*, Vol.8.No.3, pp.391-400 (Apr 1990).

[20] Postel, J., "User Datagram Protocol", RFC 768, USC/Information Sciences Institute, Marina del Rey, CA (Aug 1980).

[21] Postel, J., "Transport Control Protocol", RFC 793, USC/Information Sciences Institute, Marina del Rey, CA (Sept 1981).

[22] Root, R.W., "Design of a Multi-Media Vehicle for Social Browsing", *Proceedings Conference on Computer-Supported Cooperative Work*, Portland, OR, pp.25-38 (Sept 1988).

[23] Schooler, E.M., Casner, S.L., "A Packet-switched Multimedia Conferencing System", *ACM SIGOIS Bulletin*, Vol.10, No 1, pp.12-22 (Jan 1989).

[24] Sun Microsystems, Inc., "Remote Procedure Call: Protocol Specification", RFC 1057, Sun Microsystems, Inc., Mountain View, CA.

[25] Sun Microsystems, Inc., "External Data Representation: Protocol Specification", RFC 1014, Sun Microsystems, Inc., Mountain View, CA.

[26] Sun Microsystems, "A Socket based Interprocess Communications Tutorial", Network Programming Document, Sun Microsystems, Inc., Mountain View, CA.

[27] Swinehart, D., "The Connection Architecture for the Etherphone System", Xerox PARC, Palo Alto, CA (Dec 1990)

[28] Topolcic, C., "Experimental Internet Stream Protocol", RFC 1190, IETF CIP Working Group (Oct 1991)

[29] Vin, H.M., Swinehart, D.C., Zellweger, P.T., Rangan, V., "Multimedia Conferencing in the Etherphone Environment", Department of Computer Science and Engineering, University of California, San Diego, Technical Report Number CS91-190 (Jan 1991).

[30] Simple Network Management Protocol (SNMP), RFC 1098.